Federated Generative Learning with Foundation Models

Jie Zhang
ETH Zurich
jie.zhang@inf.ethz.ch

Xiaohua Qi
USTC
xhqi@mail.ustc.edu.cn

Bo Zhao
BAAI
zhaobo@baai.ac.cn

Abstract

Existing approaches in Federated Learning (FL) mainly focus on sending model parameters or gradients from clients to a server. However, these methods are plagued by significant inefficiency, privacy, and security concerns. Thanks to the emerging foundation generative models, we propose a novel federated learning framework, namely *Federated Generative Learning*. In this framework, each client can create text embeddings that are tailored to their local data, and send embeddings to the server. Then the informative training data can be synthesized remotely on the server using foundation generative models with these embeddings, which can benefit FL tasks. Our proposed framework offers several advantages, including **increased communication efficiency**, **robustness to data heterogeneity**, **substantial performance improvements**, and **enhanced privacy protection**. We validate these benefits through extensive experiments conducted on 12 datasets. For example, on the ImageNet100 dataset with a highly skewed data distribution, our method outperforms FedAvg by 12% in a single communication round, compared to FedAvg's performance over 200 communication rounds. We have released the code for all experiments conducted in this study[1].

1 Introduction

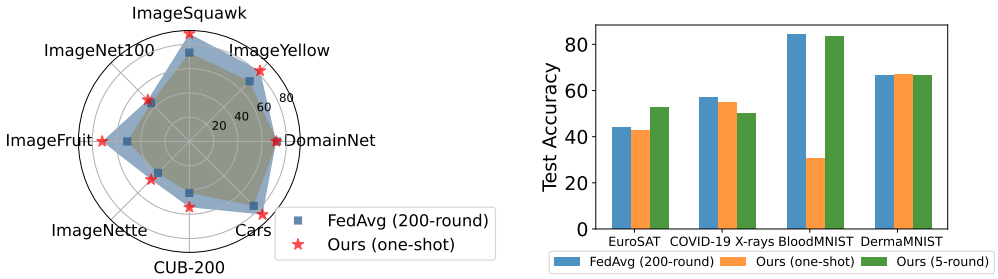
Recently, significant progress has been achieved in many learning fields by scaling up to large models, *i.e.*, BERT [8], GPT3 [2], ViT [10], CLIP [39], Stable Diffusion [42], and Web-scale datasets *i.e.*, YFCC100M [51], CC-12M [6], LAION-5B [47]. Typically, large models are first pre-trained with massive low-quality web data for basic capability, then finetuned with a small number of high-quality data, especially manually labeled data, for evoking the desired capability. Although Web data are easily accessible, high-quality training data remains scarce due to the fact that high-quality datasets are typically private or unsuitable for public release. For example, the process of labeling medical data is often costly, and the release of such data is sensitive due to safety and privacy concerns. Furthermore, raw data itself are often considered a valuable asset for numerous companies, rendering its acquisition impractical. Consequently, there is a pressing need for collaborative machine learning [11, 34, 33] that is efficient and privacy-preserving.

Federated Learning (FL) [32] has been gaining a lot of attention as a potential way to protect user privacy in distributed machine learning. When it comes to practical applications, FL systems face a few challenges that impede their real-world implementation:

1. ***High communication cost***. Current FL solutions require the transmission of model parameters or gradients between clients and the server [32, 63]. However, in the era of large models, these parameters are often in the billions or trillions, making their communication costly and even prohibitively expensive.

[1] https://github.com/zj-jayzhang/Federated_Generative_Learning

Preprint. Under review.



(a) Test accuracy on 8 datasets in one round.

(b) Results on satellite datasets and medical datasets.

Figure 1: For datasets such as subsets of ImageNet or DomainNet, our proposed method can achieve superior accuracy with only a single round of communication. In scenarios involving inherently challenging domains, including medical datasets and satellite imagery, our approach can still attain comparable performance with only five rounds of communication.

2. ***Data heterogeneity***. In FL, a fundamental challenge arises from the presence of statistical heterogeneity among local data distributions across distinct clients, which leads to performance degradation [54, 62].
3. ***Privacy and security risks***. Traditional FL involves the transmission of model parameters or gradients between clients and server. However, once these model parameters are leaked, attackers can carry out model extraction attacks [24], member inference attacks [23], and other malicious activities [67, 61], posing significant security threats to FL system [34, 30].

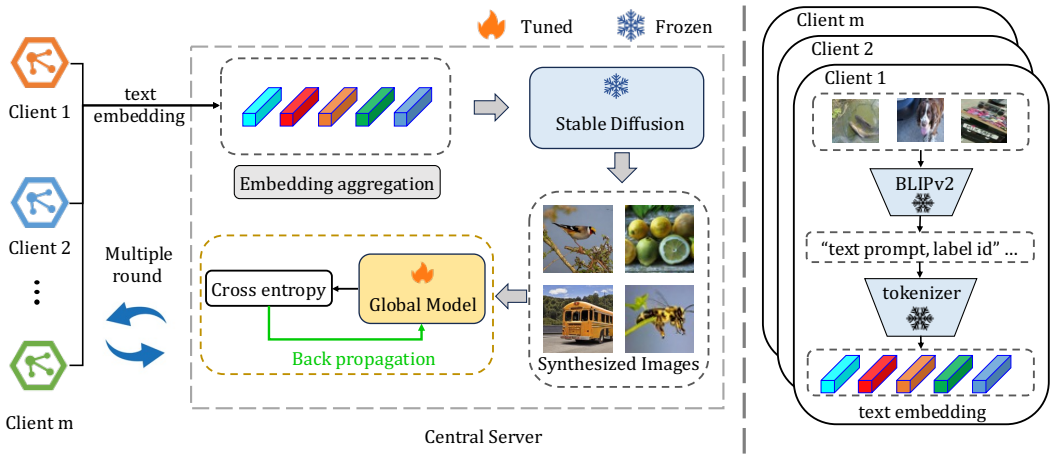
Recent advances in foundation generative models, *i.e.*, Stable diffusion [43], DALL-E2 [40], Imagen [46], and GLIDE [35], have provided a high-quality conditional text image synthesis that can be used to train models. These foundation generative models have been applied in various fields, including Computer Vision [1, 44], Speech [31, 22], and have achieved remarkable results.

In this paper, we propose a novel framework called ***Federated Generative Learning*** **(FGL)**, which leverages powerful foundation generative models, *e.g.*, Stable Diffusion [42], to synthesize high-quality training data on the server based on the text embeddings collected from clients. We propose two customized prompt generation methods based on the characteristics of the client's data, and these prompts are then used as inputs to a specific text encoder to obtain corresponding text embeddings. Once all text embeddings are collected from the clients, the server performs embedding aggregation and then synthesizes a high-quality substitute training dataset. This public synthetic dataset serves as a proxy for the clients' private data and can be used to train a global model on the server. In Figure 1(a), our trained model in a single round outperforms FedAvg with 200 communication rounds on 8 popular datasets. We further demonstrate the effectiveness of FGL by presenting results on more complex satellite dataset and three medical datasets in Figure 1(b). In summary, there are multiple benefits of FGL:

1. ***Low Communication Cost***. Compared to previous methods that rely on multi-round communication of model parameters or gradients, our method requires only one or a few communication rounds (*e.g.*, 5 rounds) between clients and the server. Despite this efficiency, our method is able to achieve performance that is on par with the existing methods.
2. ***Robust to Data Heterogeneity***. Since our method only requires clients[2] to upload text embeddings corresponding to their local training data, it allows the server to collect embeddings from all clients and synthesize all training data in one communication round. Based on the well-synthesised data, our method exhibits insensitivity to the data distribution.
3. ***Better privacy-preserving***: Previous FL methods are vulnerable to various attacks because they always transmit model parameters/gradients during the learning process. In contrast, our method only transmits text embeddings in the first round[3]. In Section 3.4, we conduct

[2]FGL requires all clients to participate in the initial round to get all text embedding first on the server. It is more suitable for cross-silo FL [17] scenarios, where the clients represent organizations or companies, and the number of clients is typically small.

[3]In subsequent rounds, FGL also transmits model parameters or gradients. However, the model memorizes very little private information after only a few rounds, which mitigates privacy risks compared to conventional FL methods based on multi-round communications. See detailed experiments on privacy analysis in Section 3.4.



(a) FGL pipeline (b) Text embedding generation

Figure 2: Training pipeline of FGL. Firstly, the text embeddings from clients are uploaded and then aggregated on the server. Then, stable diffusion is used to generate synthetic data to train the global model. Finally, the updated model are distributed to all clients. In the subfigure on the right, we present a detailed process of generating text embeddings.

a thorough privacy analysis on two aspects: (1) whether the synthetic data reveals private data information visually and (2) whether the model trained on the synthetic data is resilient against privacy attacks (e.g., membership inference attack [48]).

2 Federated Generative Learning

Framework Overview The overall framework of the proposed *Federated Generative Learning* framework is illustrated in Figure 2. Unlike traditional FL methods that transmit features, parameters, or gradients, our approach transmits text embeddings corresponding to the private data in clients to the server, thus being better privacy-preserving and communication-efficient. Then the training data is synthesized based on the aggregated text embeddings in the server with the foundation diffusion model. The synthetic training data are jointly used to train models, thus can relieve data heterogeneity problem and improve performance. Our framework has the capability to execute one-shot FL, eliminating the need for clients to train models on their local devices. Additionally, we demonstrate an extension that incorporates five communication rounds, resulting in better results. Since the generative model has never seen any private data from clients, and all synthetic data is generated and stored on the server, FGL does not necessitate additional computation resources on the client-side and does not compromise clients' privacy. The Pytorch-like pseudocode of our method is presented in Algorithm 1.

Algorithm 1 Federated Generative Learning

```
# 1) first round, create prompts and embeddings.
embed_list = []
for client in all_clients:
    # given data (x,y), generate prompts
    # class–level or instance–level
    prompts, y_s = prompts_generation(client)
    embed_list.append(text_encoder(prompt),y_s)
# 2) generate data and train global model
embeds, y_s = embeds_aggregation(embed_list)
syn_data = generative_model(embeds)
global_model = server_update(syn_data, y_s)
# 3) for 5–round communication
for com_round in [2,3,4,5]:
    local_weights = []
    for client in selected_clients:
        weight = local_update(client, global_model)
        local_weights.append(weight)
    global_model = model_averaging(local_weights)
    # finetuning for highly skewed data on server.
    if server_finetune: server_update(syn_data,
        global_model)
```

Notes: We implemented FGL in both a single communication round and five communication rounds.

2.1 Federated Learning Setting

In FL, we have K clients with their private datasets $\mathcal{D}_k = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N_k}$, where $\boldsymbol{x}_i$ is the training image, y_i is its label, $\mathcal{Y}_k$ is the label set, and N_k is the number of training samples in k-th client.

Note that the label sets of different clients may be different. The objective of the federated learning framework is to learn a model parameterized with $\boldsymbol{\theta}$ in the server that minimizes the loss on training data of all clients without access to original data: $\min_{\boldsymbol{\theta}} \frac{1}{K} \sum_{k=1}^{K} \mathbb{E}_{\boldsymbol{x} \sim \mathcal{D}_k}[l_k(\boldsymbol{\theta}; \boldsymbol{x})]$, where, l_k is the loss function for the k-th client. In this study, we explore the scenario wherein each client transfers text embeddings to the server. Subsequently, we leverage a foundational generative model on the server to generate a set of synthetic data, which can be utilized for pretraining a global model.

2.2 Text Embedding Generation and Aggregation

We investigate two types of prompt generation: class-level prompt and instance-level prompt:

- The class-level prompts are generated based on class names, providing high-level guidance to the generative model. For example, for each client, we generate a prompt like 'A photo of a {class name}' for each data. Various images with different types of noise can be generated using each prompt at the class level.
- The instance-level prompt strategy leverages prompts that are tailored for individual instances in the private dataset, which are more informative for training models. We use BLIP-v2 [25] to generate captions for each real image as the instance-level prompt (see Figure 2(b)).

Once the client generates all the prompts, these prompts can be used as inputs to a specific pretrained text encoder (e.g., from CLIP [39]) to generate all the corresponding embeddings.

Basically, given a data point (x, y), we first generate its corresponding prompt, denoted as p. Then, we create the text embedding e based on the prompt p. After receiving all (e, y) from all clients, the server aggregates them for data synthesis with foundation generative models. By default, we use the Stable Diffusion as the generative model, and in Table 3, we present the results obtained using different generative models.

2.3 Training Set Synthesis

After receiving all text embeddings, the server synthesizes every training sample $\boldsymbol{s}_i$ by prompting the pre-trained Stable Diffusion with each e_i as follows:

$$\boldsymbol{s}_i = G(\boldsymbol{z}_i, e_i) = \sqrt{\beta} \sum_{t=1}^{T} \sqrt{1-\beta^t} \cdot \frac{G_{\boldsymbol{\theta}_t}(\boldsymbol{z}_i, e_i)}{\sqrt{T}}, \tag{1}$$

where $\boldsymbol{z}_i$ is a random noise vector, e_i is the text embeddings, and $G_{\boldsymbol{\theta}_t}$ is the denoising network parameterized with $\boldsymbol{\theta}_t$ at time step t. The hyperparameter β controls the trade-off between image quality and diversity, and T is the number of diffusion steps. The inference process iteratively denoises the image then outputs the final synthetic training image. Finally, the server generates the synthetic training set $\mathcal{S} = \{(\boldsymbol{s}_i, y_i)\}_{i=1}^{N}$. Note that it is easy to synthesize more diverse training samples by combining multiple random noises with the sample prompt, thus training better models. In practice, we can adjust the number of synthetic training samples to trade-off the computational cost and performance.

2.4 Model Updating

2.4.1 One-shot Updating

We first show the efficacy of our approach in the context of one-shot federated learning [12, 60]. This involves a central server learning a global model over a network of federated devices in a single round of communication. Once the synthetic training set $\mathcal{S} = \{(\boldsymbol{s}_i, y_i)\}_{i=1}^{N}$ is obtained, we proceed to train a global model using only the synthetic data $\mathcal{S}$ on the server. Subsequently, the trained model is sent to each client as the initial model.

Robust to heterogeneous data and improved initial model. Since collecting text embeddings during the initial round of communication is not affected by the varying data distribution. Therefore, our results remain consistent in different non-IID settings for one-shot FL. In Figure 3, we have empirically demonstrated that FGL provides a well-trained initial model to each client at the first communication round. Consequently, in certain datasets, the performance of our method even exceeds that of central training after several rounds of communication (see Section 3.2 for the results).

2.4.2 Multi-round Updating

Our method can also implement multi-round communication like traditional FL methods, which can bring further performance improvement. Since the one-shot results have already shown satisfactory performance, we assume that the multi-rounds of FGL with only 5 communication rounds will also be sufficient, as additional communication rounds are considered unnecessary. We further showcase the utility of server-side synthesized data in mitigating the forgetting problem that arises after model aggregation in highly non-IID scenarios [20, 62]. Thus, for the multi-round FGL, we have implemented two versions: (1) directly utilizing the averaged model parameters without synthesized data; and (2) employing synthesized data for fine-tuning at the server-side, which is particularly useful for highly skewed data distributions. Note that the initial model is trained on the synthetic training set in the first-round communication for both two versions.

Without Synthetic Data. After the first round communication, each client receives the updated model from the server and then locally fine-tuning on it on private real training data, *i.e.*, $\boldsymbol{\theta}_k$. After locally fine-tuning, the server collects updated models from all clients for model aggregation, which is formulated as $\boldsymbol{\theta} = \frac{1}{K}\sum_{k=1}^{K}\boldsymbol{\theta}_k$, same as FedAvg. In other words, the server only aggregates models after the first round of communication. This process is repeated until the model reaching the maximum communication rounds.

With Synthetic Data. In the context of few-round communication scenarios, we find that training the aggregated model on synthesized data at the server-side can effectively mitigate the issue of forgetting [20] after aggregation, albeit at the expense of additional computational overhead. Specifically, fine-tuning the aggregated model on a synthetic training set in each communication round yields a substantial improvement in performance, particularly when dealing with highly imbalanced data distributions among clients. More results are provided in Figure 4.

Limitations of FGL. FGL relies on powerful pretrained generative models. For FL tasks that have a similar data domains to the generative model's training data, FGL can yield decent performance in a single round. However, for very challenging data domains that may not be common in the generative model's training data, such as medical data, it requires a few more rounds to achieve satisfactory results. Further improvements can consider fine-tuning the generative model locally to better adapt to the specific data domain.

3 Experimental Results

3.1 Experimental Setups

Data Partition: Follow the setting in [18, 26], we adopt two data partition settings, namely, label distribution skew and feature distribution skew:

- **Label Distribution Skew**: Following in [62], in which the label distributions varies on different clients, we employ the Dirichlet distribution $p \sim Dir(\beta)$ to simulate imbalanced label distributions. The hyper-parameter β controls the degree of label imbalance, where a smaller value of β indicates a more skewed label distribution.
- **Feature Distribution Skew**: In this setting, clients share the same label space while having a different feature distribution, which has been extensively studied in previous work [28, 65, 57, 11]. We perform the classification task on natural images sourced from DomainNet [38], which consists of diverse distributions of natural images from six distinct data sources.

Baselines: In our experiments, we select the popular FedAvg [32] method and centralized training as the baselines by default[4]. Assume that we have a total of 5 clients[5] and that every client participates in communication. For both centralized training and federated learning, the local learning rate is set to 0.01, and we utilize the SGD optimizer with a momentum of 0.9. During the FedAvg training

[4]We also compared FGL with other baselines, *e.g.*, Moon [27], Fedopt [41], as shown in Appendix Table 8.

[5]Further results on 50 and 100 clients can be found in Table 4. Varying the number of clients does not have a significant impact on our method, as long as the server can obtain all the text embeddings and subsequently synthesize well-generated data.

Table 1: Performance comparison among different methods on 7 datasets. The improvement $_{\uparrow}$ is compared to FedAvg IID results.

Dataset	FedAvg			Ours (one-shot)	Ours (5-round)			Centralized
	$\beta = 0.01$	$\beta = 0.5$	IID		$\beta = 0.01$	$\beta = 0.5$	IID	
ImageNette	51.6	75.0	79.2	$85.2_{\uparrow 6.0}$	82.8	94.0	**95.6**$_{\uparrow 16.4}$	92.2
ImageFruit	29.0	51.2	55.6	$71.8_{\uparrow 16.2}$	67.2	80.2	**83.2**$_{\uparrow 27.6}$	78.2
ImageYellow	50.6	70.2	74.6	$82.4_{\uparrow 7.8}$	79.4	91.0	**94.8**$_{\uparrow 20.2}$	90.8
ImageSquawk	49.6	73.2	79.8	$88.8_{\uparrow 9.0}$	90.0	95.0	**95.6**$_{\uparrow 15.8}$	92.4
ImageNet100	36.3	44.6	49.4	$48.4_{\downarrow 1.0}$	70.1	74.9	**80.1**$_{\uparrow 30.7}$	77.0
CUB-200	35.0	36.6	36.6	$44.6_{\uparrow 8.0}$	67.7	71.9	**73.3**$_{\uparrow 37.3}$	48.3
Stanford Cars	-	42.4	44.5	$54.23_{\uparrow 9.7}$	85.4	88.0	**88.8**$_{\uparrow 44.3}$	64.7

process, each client performs local updates for 5 epochs, and the communication round is set to 200. The centralized training consists of 120 rounds of iterations.

Implementation: We employ the class-level prompt by default, which does not directly utilize local data information, thus providing enhanced privacy protection. We use Stable Diffusion v2-1-base model to construct synthetic data. We evaluate the performance of our method in scenarios of one-round (*i.e.*, **Ours (one-shot)**) and five-round communications (*i.e.*, **Ours (5-round)**). In the one-round communication scenario, we train the model for 120 epochs on the server. In the five-round communication scenario, we implement two variations: (1) **Ours (5-round)**: based on the model trained in the first round, we perform four additional rounds of communication using the FedAvg algorithm. (2) **Ours (5-round-syn)** : for extreme data distribution skew scenario, we further finetune the aggregated model using the synthetic dataset generated on the server during the first round for all five epochs. Please refer to **Appendix** 6.1 for more details on implementation and datasets.

3.2 Results for Label Distribution Skew

To evaluate the efficacy of our method on datasets with label distribution skew, we conduct experiments on five subsets of 224×224 ImageNet [45]. Firstly, following [16, 5], we do experiments on four datasets with 10 categories each, namely the coarse-grained ImageNette and ImageYellow, and the fine-grained ImageFruit and ImageSquawk. We further conducted experiments on ImageNet100, involving 100 categories. Also, we conducted experiments on two fine-grained image classification datasets, namely CUB-200 [53], Stanford Cars [19]. We simulate three distinct dataset distributions, *i.e.*, IID, non-IID ($\beta = 0.5$) and highly skewed distribution with $\beta = 0.01$.

The overall experimental results are shown in Table 1. It is evident that our method with one-round communication outperforms the FedAvg method with 200 rounds of communication, by 6.0%, 16.2%, 7.8% and 9.0% on the four 10-category datasets in IID setting. Notably, our method is completely insensitive to data distribution in the first round. Hence, under extreme data distribution, *i.e.*, $\beta = 0.01$, our method surpasses FedAvg by 33.6%, 42.8%, 31.8% and 39.2% on the four 10-category ImageNet subsets respectively. Also, our method outperforms FedAvg by a minimum of 30% on ImageNet100, CUB-200, and Stanford Cars, after 5 rounds of communication, and even exceeds the performance of centralized training on these datasets.

Why does training with synthetic data yield better results? CUB-200 is a challenging dataset consisting of 200 bird species with 11,788 images, and Cars contains 16,185 images belonging to 196 classes of cars. The size of these fine-grained recognition datasets is typically smaller compared to general image classification datasets. In previous work [66, 9], a common practice is to utilize a pretrained model that has been trained on the ImageNet dataset. In this study, we present two approaches: training the model from scratch and loading a pretrained ResNet34 model. Note that ImageNet has approximately 1.2M training data, but we only use about 0.18M images. We present the accuracy gap between FGL and FedAvg in Figure 3. It is evident that by directly loading a model pre-trained on ImageNet, the accuracy gap can be significantly reduced.

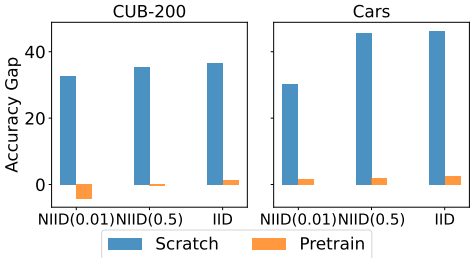


Figure 3: Accuracy gap when loading a pre-trained model or training from scratch.

Takeaways: In the first round of communication, FGL generates a set of synthetic data to train a model, which serves as an excellent initial model. Unlike 'out-of-domain' datasets such as ImageNet, this smaller 'in-domain' synthetic data exhibits remarkable performance in FL tasks.

Results on Highly Skewed Data. Table 1 shows that when the data distribution is extremely skewed ($\beta = 0.01$), Ours (5-round) does not outperform Ours (one-shot) on Imagenette, ImageFruit, and ImageYellow datasets. We attribute this phenomenon to the fact that in case of highly skewed data distribution, more rounds of communication are required for the models to converge gradually [63, 54]. To achieve better results within five rounds, we perform fine-tuning on the aggregated model using the synthesized dataset from the first round on the server for 5 epochs before distributing the updated model. The results are presented in Figure 4, which clearly demonstrate that fine-tuning on our synthesized dataset can significantly enhance model performance even with extreme data distribution.

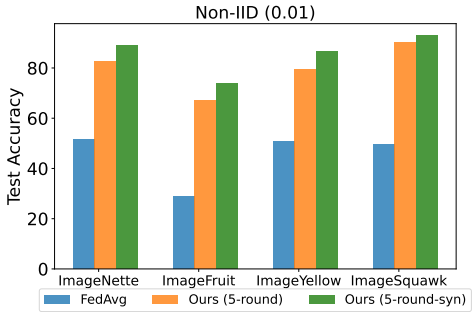


Figure 4: Accuracy on highly skewed data.

Number of Synthetic Data. By comparing the results of $1.3k$ synthetic images per class in Appendix (Fig. 11) and $20k$ synthetic images per class Table 1, we find the performance of our method can significantly improve by synthesizing more training samples, *i.e.*, from 73.2% to 85.2% on the Imagenette dataset. We further study the influence of synthetic image number and model performance in Figure 5 on four 10-category datasets. We synthesize more images per class by integrating the prompts and more random noises. Obviously, as the number of images per class increases from 2k to 20k, the test accuracy improves consistently.

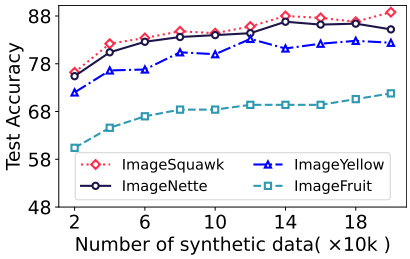


Figure 5: Varying synthetic data volume.

Results on Medical and Satellite Datasets We further show the performance of our method on datasets with challenging domains such as medical datasets and satellite images to explore the limits of our approach. Figure 1(b) illustrates the performance of FedAvg and our proposed method on the EuroSAT [15] satellite dataset as well as three medical datasets, namely COVID-19 X-rays [7], BloodMNIST and DermaMNIST [55]. Our method demonstrates comparable performance to FedAvg (200-round) after 5 communication rounds on both medical and satellite datasets. Figure 6 presents the visualization of the synthetic and real data, highlighting the difficulties encountered in generating data for these challenging domains. The capability of the generative model is somewhat diminished in these scenarios. Please refer to the Appendix (Table 9) for more detailed results under different non-IID settings.

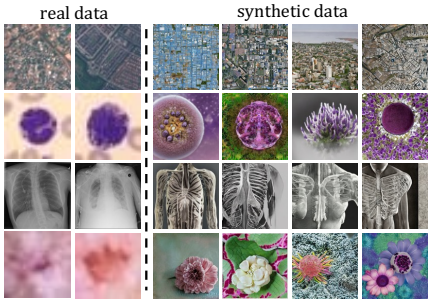


Figure 6: Visualization of real data and synthetic data, where the datasets EuroSAT, BloodMNIST, COVID-19 X-rays, and DermaMNIST are displayed from top to bottom.

Takeaways: The quality of synthetic data may be influenced by the domain discrepancy between the local training data and the pretraining data used for the foundation model. This discrepancy ultimately limits the performance of FGL. One potential solution is to finetune the foundation model using local data, however, this could potentially raise privacy concerns. We leave it for future research.

3.3 Results for Feature Distribution Skew

To simulate the scenario of feature distribution skew, we select 10 categories from the DomainNet dataset to conduct experiments . Each client is assigned a specific domain, and we have a total of 6

Table 2: Performance for feature distribution skew. Each client hosts data from a specific domain of DomainNet dataset.

Method	Clipart	Infograph	Painting	Quickdraw	Real	Sketch	Average
FedAvg	80.97	41.90	57.33	78.93	80.56	70.06	72.30
Centralized	81.37	50.82	60.63	92.46	82.20	73.93	78.10
Ours (one-shot)	83.40$_{\uparrow 2.43}$	49.58$_{\uparrow 7.68}$	76.88$_{\uparrow 19.55}$	51.80$_{\downarrow 27.13}$	87.06$_{\uparrow 6.5}$	81.10$_{\uparrow 11.04}$	71.59$_{\downarrow 0.71}$
Ours (5-round)	**90.89**$_{\uparrow 9.92}$	**61.61**$_{\uparrow 19.71}$	**79.52**$_{\uparrow 22.19}$	**81.13**$_{\uparrow 2.2}$	**91.13**$_{\uparrow 10.57}$	**90.20**$_{\uparrow 20.14}$	**84.05**$_{\uparrow 11.75}$

clients participating in FL. To ensure an adequate amount of data, we synthesize 3,500 samples for each class within each domain, resulting in a cumulative dataset of $210k$ samples. Table 2 presents the performance of various methods on six domains respectively and their average accuracy. It shows that in the one-shot communication scenario, our method outperforms FedAvg by 2% to 19% in five domains, but exhibits notably poor performance in the Quickdraw domain. To investigate the underlying reason, we visualize the synthetic and real data in Appendix 6.2.1. It becomes apparent that this performance decline is attributed to the difficulty for diffusion model to synthesize images that align with Quickdraw domain when using the class-level prompt, *i.e.*, "A black and white drawing of a {class name}". We present the results of the generative model on medical and satellite datasets in Figure 6, which also demonstrate the limited synthetic capability of the generative model in challenging domains.

However, when implementing a five-round communication experiment, our method demonstrates a 2.2% performance improvement over FedAvg specifically on the Quickdraw domain, and an overall performance improvement of 11.75%. Interestingly, our method even surpasses the performance of the centralized training models by 5.95%. We provide further results on each domain in Appendix 6.2.2, and more visualizations of synthetic data on DomainNet and ImageNet are shown in Appendix 6.2.3.

3.4 Privacy Analysis

Membership Inference Attack (MIA). The objective of MIA is to examine whether a specific data point belongs to the training set used to train a machine learning model. Given that Ours (one-shot) does not depend on real training data, it is reasonable to assume that it may not encounter any privacy leakage. However, since Ours (5-round) and FedAvg both utilize private data for training, we present the MIA results on ImageNette using the low false-positive rate regime, as recommended by the state-of-the-art Likelihood Ratio Attack (LiRA) [3]. As show in Figure 7, when employing the LiRA against models trained using private data (*i.e.*, FedAvg) and synthetic data (*i.e.*, Ours (5-round)), the latter exhibits a stronger defense against membership inference attacks. This can be attributed to the fact that our model, trained on synthetic data, exhibits minimal information leakage.

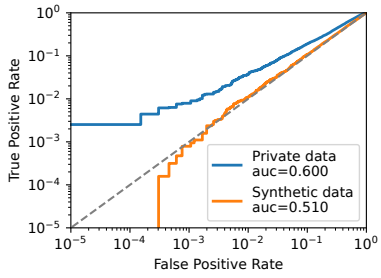


Figure 7: Attack results under LiRA for models trained on real data and synthetic data. A total of 32 shadow models were trained, ensuring that each sample was trained on half of these models.

Detecting Content Replication and Memorization. Previous research [4, 59, 52, 50, 49] has indicated that diffusion models store and reproduce specific images from their training dataset during the generation process. Although our training process (with class-level prompts) does not access the private data of clients, we still discuss the potential privacy risks that may arise. Follow the setting in [49], we conduct image retrieval experiments, which allows us to compare the synthetic images with the original training images and detect any instances of content duplication. We perform a quantitative analysis on 1000 synthetic images across four datasets. For each synthetic image, we search the training set by computing the Cosine similarity between its feature and features of real training images. Figure 8 showcases the top 2 most similar images

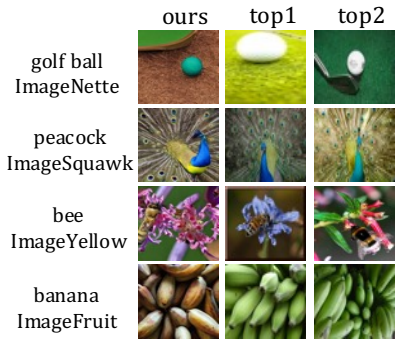


Figure 8: Retrieving similar real images for each synthetic image.

Table 3: Ablation study on the generative model used in FGL.

Method	one-shot	5-round, $\beta = 0.01$	5-round, $\beta = 0.5$	IID	Centralized
Ours w/ SD	**85.2**	**82.8**	**94.1**	**95.6**	92.2
Ours w/ Glide	79.0	76.2	89.4	89.4	92.2
Ours w/ Dit	76.2	74.6	90.2	92.8	92.2
FedAvg (120-round)	-	51.6	75.1	79.2	92.2

from each of the four datasets. These images exhibit no noteworthy similarities in background and foreground, which verifies FGL doesn't compromise privacy of the client's private data.

3.5 Ablation Study

Varying the Generative Models. To investigate the impact of various generative models on the results, we followed the setting in [29]. Our experiments primarily focus on three prevalent conditional diffusion models: DiT [37], GLIDE [36], and Stable Diffusion. We use these off-the-shelf models to generate synthetic images. Specifically, for GLIDE and Stable Diffusion, the prompt was configured as "a photo of {label name}, real-world images, high resolution". For DiT, the input comprised the label ID corresponding to the ImageNet1k dataset. The images synthesized by DiT and GLIDE are of dimensions 256x256, whereas those produced by Stable Diffusion are of dimensions 512x512. As shown in Table 3, even when we vary the foundation models used in our method, FGL consistently outperforms FedAvg by a significant margin.

Results on more clients and more baselines To demonstrate the scalability of our method to a larger number of clients, we extended our analysis to include the results obtained from the ImageNette dataset with 50 and 100 clients. As depicted in Table 4 (see the appendix), our method continues to exhibit superior performance compared to FedAvg across all scenarios. Additionally, the improvements achieved by our method remain significant. Also, we have compared the two popular FL methods, Moon [27] and Fedopt [41]. We conducted experiments on the ImageNette and ImageNet100 datasets, considering a scenario with 50 clients under non-IID settings ($\beta = 0.5$). As shown in the Table 8 (see the appendix), our method still outperforms other FL approaches.

4 Related Work

Foundation Generative Models. Large generative models, such as Stable Diffusion [43], DALL-E2 [40], Imagen [46], and GLIDE [35], have recently emerged as an off-the-shelf tool for high-quality and real-looking image generation conditioned on text prompts. A few works have explored the usage of synthetic images as training data. For example, He et al. [14] show that synthetic data generated by diffusion models can improve pretraining, zero-shot, and few-shot image classification performance. Li [29] demonstrate that synthetic data generated by conditional diffusion models can be used for knowledge distillation without original data. Zhou [64] synthesize better images for model training with stable diffusion by implementing diffusion inversion.

Foundation Models in FL. The foundation generative models are still under-explored in federated learning, though there exist a few related works that study foundation models in federated learning. The most similar one is [56], which takes advantage of the diffusion model in the server to synthesize training samples that complied to the distributions of domain-specific features from clients. Yu et al. [58] introduce federated learning into foundation model training for training foundation models collaboratively with private data. Like traditional FL methods, they also transmit model parameters between servers and clients. Based on the shared CLIP model, Guo et al. [13] transmit the small number of updated parameters of the prompt learner from clients to the server to reduce the communication cost.

5 Conclusion

In this work, we introduce a pioneering framework, named *Federated Generative Learning*, which transmits prompts associated with distributed training data between clients and the server. By leveraging foundation generative models, informative training data can be synthesized remotely using received prompts that contain minimal privacy. The proposed framework exhibits several noteworthy advantages, including improved communication efficiency, better resilience to distribution shift, substantial performance gains, and enhanced privacy protection.

References

[1] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22563–22575, 2023.

[2] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[3] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramer. Membership inference attacks from first principles. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 1897–1914. IEEE, 2022.

[4] Nicholas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwag, Florian Tramer, Borja Balle, Daphne Ippolito, and Eric Wallace. Extracting training data from diffusion models. *arXiv preprint arXiv:2301.13188*, 2023.

[5] George Cazenavette, Tongzhou Wang, Antonio Torralba, Alexei A Efros, and Jun-Yan Zhu. Dataset distillation by matching training trajectories. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4750–4759, 2022.

[6] Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12M: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *CVPR*, 2021.

[7] Muhammad EH Chowdhury, Tawsifur Rahman, Amith Khandakar, Rashid Mazhar, Muhammad Abdul Kadir, Zaid Bin Mahbub, Khandakar Reajul Islam, Muhammad Salman Khan, Atif Iqbal, Nasser Al Emadi, et al. Can ai help in screening viral and covid-19 pneumonia? *Ieee Access*, 8:132665–132676, 2020.

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[9] Qishuai Diao, Yi Jiang, Bin Wen, Jia Sun, and Zehuan Yuan. Metaformer: A unified meta framework for fine-grained recognition. *arXiv preprint arXiv:2203.02751*, 2022.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[11] Xuan Gong, Abhishek Sharma, Srikrishna Karanam, Ziyan Wu, Terrence Chen, David Doermann, and Arun Innanje. Preserving privacy in federated learning with ensemble cross-domain knowledge distillation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11891–11899, 2022.

[12] Neel Guha, Ameet Talwalkar, and Virginia Smith. One-shot federated learning. *arXiv preprint arXiv:1902.11175*, 2019.

[13] Tao Guo, Song Guo, Junxiao Wang, and Wenchao Xu. Promptfl: Let federated participants cooperatively learn prompts instead of models–federated learning in age of foundation model. *arXiv preprint arXiv:2208.11625*, 2022.

[14] Ruifei He, Shuyang Sun, Xin Yu, Chuhui Xue, Wenqing Zhang, Philip Torr, Song Bai, and Xiaojuan Qi. Is synthetic data from generative models ready for image recognition? *arXiv preprint arXiv:2210.07574*, 2022.

[15] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7):2217–2226, 2019.

[16] Jeremy Howard. A smaller subset of 10 easily classified classes from imagenet, and a little more french, 2019. *URL https://github. com/fastai/imagenette*.

[17] Chao Huang, Jianwei Huang, and Xin Liu. Cross-silo federated learning: Challenges and opportunities. *arXiv preprint arXiv:2206.12949*, 2022.

[18] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

[19] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE international conference on computer vision workshops*, pages 554–561, 2013.

[20] Gihun Lee, Minchan Jeong, Yongjin Shin, Sangmin Bae, and Se-Young Yun. Preservation of the global knowledge by not-true distillation in federated learning. *arXiv preprint arXiv:2106.03097*, 2021.

[21] Shiye Lei, Hao Chen, Sen Zhang, Bo Zhao, and Dacheng Tao. Image captions are natural prompts for text-to-image models. *arXiv preprint arXiv:2307.08526*, 2023.

[22] Alon Levkovitch, Eliya Nachmani, and Lior Wolf. Zero-shot voice conditioning for denoising diffusion tts models. *arXiv preprint arXiv:2206.02246*, 2022.

[23] Jiacheng Li, Ninghui Li, and Bruno Ribeiro. Effective passive membership inference attacks in federated learning against overparameterized models. In *The Eleventh International Conference on Learning Representations*, 2023.

[24] Jingtao Li, Adnan Siraj Rakin, Xing Chen, Li Yang, Zhezhi He, Deliang Fan, and Chaitali Chakrabarti. Model extraction attacks on split federated learning. *arXiv preprint arXiv:2303.08581*, 2023.

[25] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.

[26] Qinbin Li, Yiqun Diao, Quan Chen, and Bingsheng He. Federated learning on non-iid data silos: An experimental study. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pages 965–978. IEEE, 2022.

[27] Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10713–10722, 2021.

[28] Xiaoxiao Li, Meirui Jiang, Xiaofei Zhang, Michael Kamp, and Qi Dou. Fedbn: Federated learning on non-iid features via local batch normalization. *arXiv preprint arXiv:2102.07623*, 2021.

[29] Zheng Li, Yuxuan Li, Penghai Zhao, Renjie Song, Xiang Li, and Jian Yang. Is synthetic data from diffusion models ready for knowledge distillation? *arXiv preprint arXiv:2305.12954*, 2023.

[30] Pengrui Liu, Xiangrui Xu, and Wei Wang. Threats, attacks and defenses to federated learning: issues, taxonomy and perspectives. *Cybersecurity*, 5(1):1–19, 2022.

[31] Zhijun Liu, Yiwei Guo, and Kai Yu. Diffvoice: Text-to-speech with latent diffusion. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1–5. IEEE, 2023.

[32] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[33] Viraaji Mothukuri, Reza M Parizi, Seyedamin Pouriyeh, Yan Huang, Ali Dehghantanha, and Gautam Srivastava. A survey on security and privacy of federated learning. *Future Generation Computer Systems*, 115:619–640, 2021.

[34] Truc Nguyen and My T Thai. Preserving privacy and security in federated learning. *arXiv preprint arXiv:2202.03402*, 2022.

[35] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

[36] Alexander Quinn Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob Mcgrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. In *International Conference on Machine Learning*, pages 16784–16804. PMLR, 2022.

[37] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

[38] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1406–1415, 2019.

[39] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[40] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

[41] Sashank Reddi, Zachary Charles, Manzil Zaheer, Zachary Garrett, Keith Rush, Jakub Konečnỳ, Sanjiv Kumar, and H Brendan McMahan. Adaptive federated optimization. *arXiv preprint arXiv:2003.00295*, 2020.

[42] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10684–10695, June 2022.

[43] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[44] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22500–22510, 2023.

[45] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[46] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022.

[47] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022.

[48] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership inference attacks against machine learning models. In *IEEE symposium on security and privacy*, 2017.

[49] Gowthami Somepalli, Vasu Singla, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Diffusion art or digital forgery? investigating data replication in diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6048–6058, 2023.

[50] Gowthami Somepalli, Vasu Singla, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Understanding and mitigating copying in diffusion models. *arXiv preprint arXiv:2305.20086*, 2023.

[51] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *Communications of the ACM*, 59(2):64–73, 2016.

[52] Gerrit van den Burg and Chris Williams. On memorization in probabilistic deep generative models. *Advances in Neural Information Processing Systems*, 34:27916–27928, 2021.

[53] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011.

[54] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H Vincent Poor. Tackling the objective inconsistency problem in heterogeneous federated optimization. *Advances in neural information processing systems*, 33:7611–7623, 2020.

[55] Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. Medmnist v2-a large-scale lightweight benchmark for 2d and 3d biomedical image classification. *Scientific Data*, 10(1):41, 2023.

[56] Mingzhao Yang, Shangchao Su, Bin Li, and Xiangyang Xue. Exploring one-shot semi-supervised federated learning with a pre-trained diffusion model. *arXiv preprint arXiv:2305.04063*, 2023.

[57] Chun-Han Yao, Boqing Gong, Hang Qi, Yin Cui, Yukun Zhu, and Ming-Hsuan Yang. Federated multi-target domain adaptation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1424–1433, 2022.

[58] Sixing Yu, J Pablo Muñoz, and Ali Jannesari. Federated foundation models: Privacy-preserving and collaborative learning for large models. *arXiv preprint arXiv:2305.11414*, 2023.

[59] Eric Zhang, Kai Wang, Xingqian Xu, Zhangyang Wang, and Humphrey Shi. Forget-me-not: Learning to forget in text-to-image diffusion models. *arXiv preprint arXiv:2303.17591*, 2023.

[60] Jie Zhang, Chen Chen, Bo Li, Lingjuan Lyu, Shuang Wu, Shouhong Ding, Chunhua Shen, and Chao Wu. Dense: Data-free one-shot federated learning. *Advances in Neural Information Processing Systems*, 35:21414–21428, 2022.

[61] Jie Zhang, Bo Li, Chen Chen, Lingjuan Lyu, Shuang Wu, Shouhong Ding, and Chao Wu. Delving into the adversarial robustness of federated learning. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence*, AAAI'23/IAAI'23/EAAI'23. AAAI Press, 2023.

[62] Jie Zhang, Zhiqi Li, Bo Li, Jianghe Xu, Shuang Wu, Shouhong Ding, and Chao Wu. Federated learning with label distribution skew via logits calibration. In *International Conference on Machine Learning*, pages 26311–26329. PMLR, 2022.

[63] Yue Zhao, Meng Li, Liangzhen Lai, Naveen Suda, Damon Civin, and Vikas Chandra. Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*, 2018.

[64] Yongchao Zhou, Hshmat Sahak, and Jimmy Ba. Training on thin air: Improve image classification with generated data. *arXiv preprint arXiv:2305.15316*, 2023.

[65] Guogang Zhu, Xuefeng Liu, Shaojie Tang, and Jianwei Niu. Aligning before aggregating: Enabling cross-domain federated learning via consistent feature extraction. In *2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS)*, pages 809–819. IEEE, 2022.

[66] Haowei Zhu, Wenjing Ke, Dong Li, Ji Liu, Lu Tian, and Yi Shan. Dual cross-attention learning for fine-grained visual categorization and object re-identification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4692–4702, 2022.

[67] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. *Advances in neural information processing systems*, 32, 2019.

Table 4: Results on more clients.

Client	FedAvg $\beta = 0.5$	FedAvg (IID)	Ours (one-shot)	Ours (5-round) $\beta = 0.5$	IID	Centralized
5	75	79.2	85.2	94	95.6	92.2
50	72	77.0	85.2	93.8	91.2	92.2
100	70.0	67.2	85.2	92.8	93.2	92.2

Table 5: Examples of prompt generation patterns for Class-Level and Instance-Level prompting

Prompt Type	**Dataset**	**Prompt Template**	**Prompt Example**
Class Level	ImageNet Subset	label + ', real world images , high resolution'	tench, real world images, high resolution
Class Level	DomainNet Subset	label + style	an airplane, Sketch drawing with only one object in the picture
Instance Level	ImageNet Subset	label + ', ' + image caption + ', real world images, high resolution.'	tench, Tinca tinca, a man kneeling down holding a fish in the grass

6 Appendix

6.1 Experiments Setting

6.1.1 Prompt Generation for Synthetic Data in ImageNet and DomainNet

In this section, we present the configurations for prompt generation when synthesizing data for the ImageNet and DomainNet datasets. As depicted in Table 5, for class-level prompt generation on ImageNet-like datasets, the prompt template consists of the label followed by ", real-world images, high resolution." Examples of generated prompts include "tench, real world images, high resolution" and "English springer, real world images, high resolution." On DomainNet Subset datasets, the prompt template comprises the label and style, where the label represents the category name, and the style describes the domain. Examples of generated prompts in this context are "an airplane, Sketch drawing with only one object in the picture" and "an airplane, real world images, high resolution, with only one object in the picture."

For instance-level prompt generation on ImageNet Subset datasets, the prompt template consists of the label followed by ", " and the image caption followed by ", real-world images, high resolution." Here, the label represents the category name, and image caption corresponds to the textual description of the image generated by BLIPv2. Examples of generated prompts include "tench, Tinca tinca, a man kneeling down holding a fish in the grass" and "tench, Tinca tinca, a man kneeling down holding a large fish in the water."

6.1.2 Dataset Description

In this section, we provide a detailed description of the datasets used in our experiments. The datasets include the DomainNet Subset, lmageNette, lmageFruit, lmageYellow, lmageSquawk, ImageNet100, Eurosat, COVID-19 X-rays, BloodMNIST and DermaMNIST.

DomainNet Subset: This subset is selected from the DomainNet dataset and consists of ten categories spanning six different domains. Refer to Table 6 for detailed class and domain names.

ImageNet Subset: These datasets are subsets extracted from ImageNet, including lmageNette, lmageFruit, lmageYellow, lmageSquawk, and ImageNet100. Refer to Table 7 for details on class names and class IDs.

Table 6: Detailed description of the DomainNet Subset Dataset

Description	# class	Class name	Domain name
10 classes from DomainNet	10	airplane, clock, axe, basketball, bicycle, bird, strawberry, flower, pizza, bracelet	Clipart, Infograph, Painting, Quickdraw, Real, Sketch

Table 7: Detailed description of the ImageNet Subset Dataset

Dataset	Description	# Class	Class name	Class id
lmageNette	10 class from ImageNet	10	(tench, English springer, cassetteplayer, chain saw, church, Frenchhorn, garbage truck, gas pump, golfball, parachute)	(0, 217, 482, 491, 497, 566, 569,571, 574, 701)
lmageFruit	10 class from ImageNet	10	(pineapple, banana, strawberry, orange, lemon, pomegranate, fig, bell pepper, cucumber, green apple)	(953, 954, 949, 950, 951, 957, 952, 945, 943, 948)
lmageYellow	10 class from ImageNet	10	(bee, ladys slipper, banana, lemon, corn, school bus, honeycomb, lion, garden spider, goldfinch)	(309,986, 954, 951, 987, 779, 599, 291, 72, 11)
lmageSquawk	10 class from ImageNet	10	(peacock, flamingo, macaw, pelican, king penguin, bald eagle, toucan, ostrich, black swan, cockatoo)	(84, 130, 88, 144, 145, 22, 96, 9, 100, 89)
ImageNet100	100 class from ImageNet	100	- -	- -

Eurosat: The Eurosat dataset [15], derived from Sentinel-2 satellite images with 13 spectral bands, consists of 10 classes, totaling 27,000 labeled and geo-referenced images. Official images are of size 64×64, whereas our synthesized data is 512×512. To enhance diversity, we randomly crop the synthesized data to 64×64, 128×128, or 224×224, followed by resizing to 64×64 for training.

COVID-19 X-rays: The COVID-19 X-rays dataset [7], categorized into COVID-19, normal, and viral pneumonia classes, follows the COVIDx-8A version with 5,585 training images and 400 test images. During training, images are resized to 256×256 and subjected to random resized cropping to 224×224.

MedMNIST v2 Subset: Both the BloodMNIST and DermaMNIST datasets are constituents of MedMNIST v2 [55], a comprehensive MNIST-like compilation of standardized biomedical images.The DermaMNIST dataset originates from HAM10000, a vast collection of dermatoscopic images of common pigmented skin lesions from multiple sources. The dataset comprises 10,015 dermatoscopic images categorized into seven different diseases, structured as a multi-class classification task. We follow the official partitioning of training, validation, and test sets with a ratio of 7:1:2. The image dimensions are uniform at $3 \times 28 \times 28$, and for synthesized data, we resize to $3 \times 28 \times 28$ for training.The BloodMNIST dataset is derived from individual normal cells captured from individuals devoid of infection, hematologic or oncologic diseases, and without any pharmacologic treatment at the time of blood collection. Comprising a total of 17,092 images distributed across eight classes, we adopt the official partitioning with a ratio of 7:1:2 for training, validation, and test sets. The image dimensions are consistently $3 \times 28 \times 28$, and for synthesized data, we resize to $3 \times 28 \times 28$ during training.

6.2 Additional Experiments

6.2.1 Synthetic Data Visualization on Quickdraw Domain

To investigate the reasons behind the notably poor performance in the Quickdraw domain, we visualize both synthetic and real data for the Painting and QuickDraw domains of the DomainNet dataset in Figure 9. The first and second rows correspond to real data and synthetic data for the Painting Domain, respectively. It is evident that the synthetic data closely resembles the real data in style, which results in the model performing well in this domain. The third and fourth rows represent real data and synthetic data for the Quickdraw Domain, respectively. Notably, there is a substantial

stylistic difference between the synthetic data and real data in this domain, providing an explanation for the poor model performance.

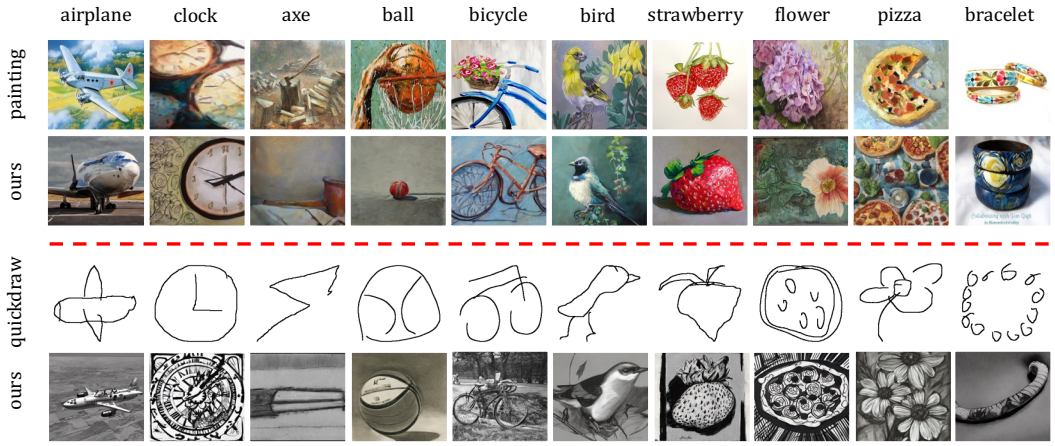


Figure 9: Visualization of original and synthetic data on the Painting and QuickDraw domains of DomainNet dataset. Obviously, it is easy for diffusion model to synthesize painting-like images, while challenging to synthesize images with QuickDraw style.

6.2.2 Testing Accuracy on Each Domain.

In this section, we compare the performance of the "Centralized" and "Ours (One-shot)" methods on the DomainNet dataset across different domains during the training process. The experimental results, as shown in Figure 10, demonstrate that, except for the "Quickdraw" domain, Ours (One-shot) outperforms the Centralized Training in five domains. Particularly, in the "Painting" domain, our method achieves a significant performance improvement. We provide further explanation about the performance gap in "Quickdraw" domain through visualization in the following section.

6.2.3 Visualization on DomainNet and ImageNet

In this section, we provide comprehensive visualizations of the synthetic data generated from the ImageNet and DomainNet datasets. Figure 12 showcases visualizations of synthetic data from four distinct subsets of the ImageNet dataset. Each pair of rows corresponds to one of these subsets,

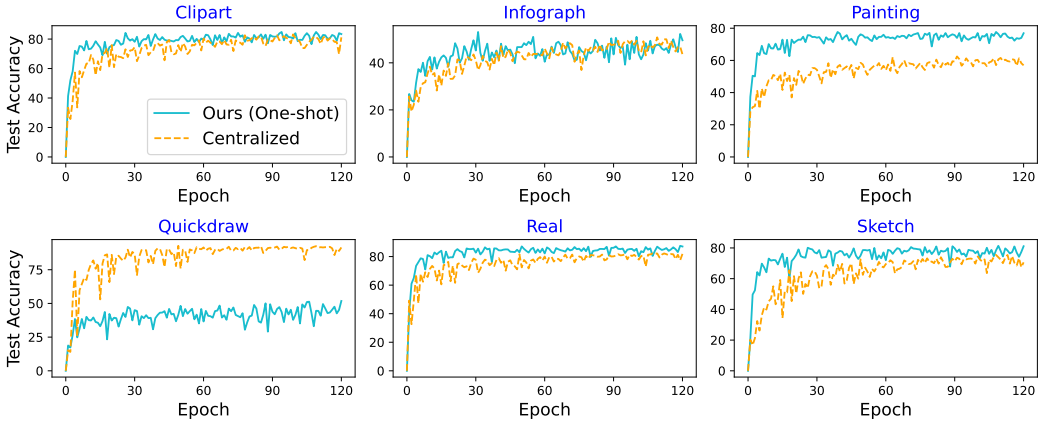


Figure 10: Performance comparison between Ours (One-shot) and the Centralized Training on six domains of DomainNet dataset.

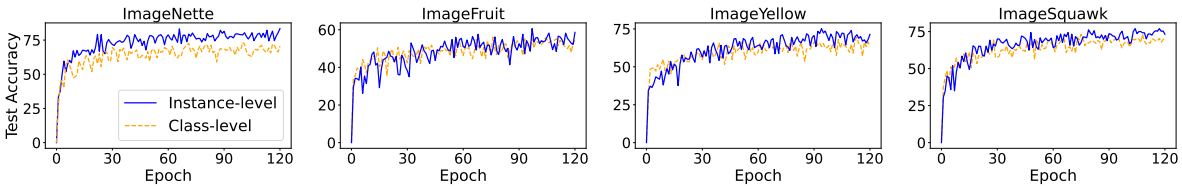


Figure 11: Accuracy of class-level prompt and instance-level prompt on four datasets. We synthesize 1300 images per class. After using instance-level prompts, the best accuracy improved by 10.2%, 4.6%, 5.8%, and 3.8% respectively.

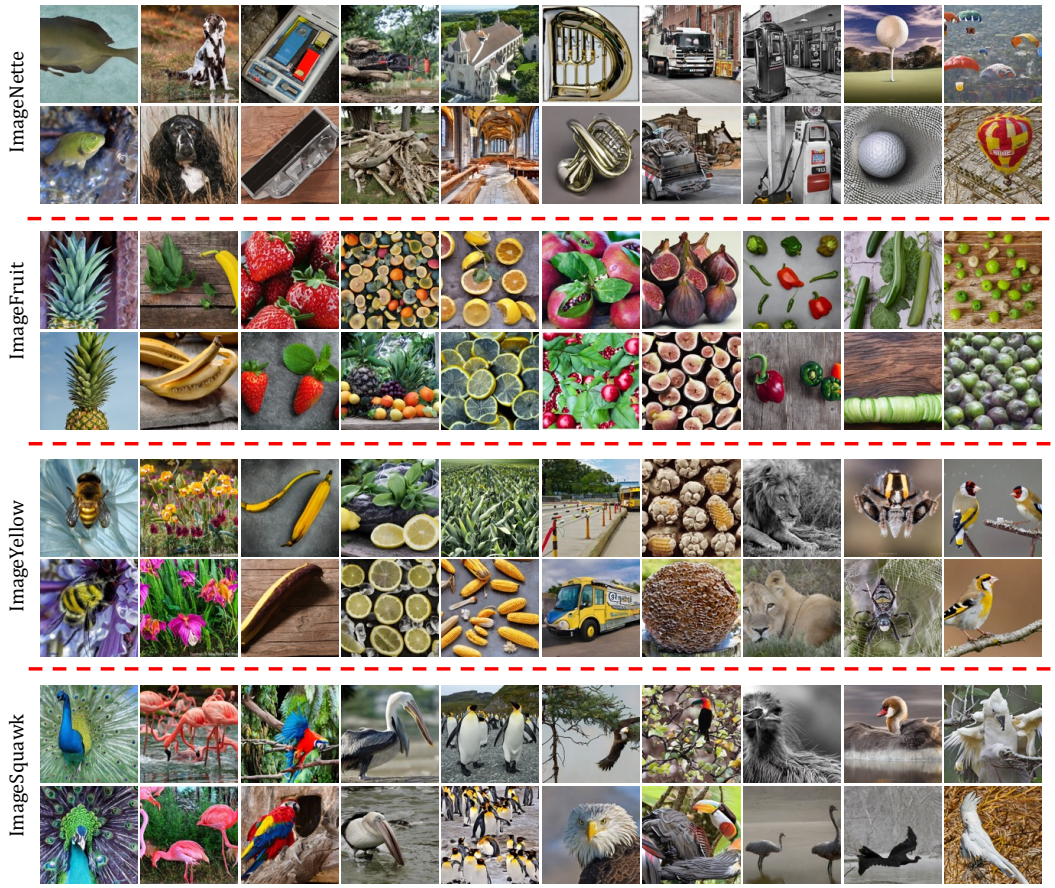


Figure 12: Visualization of synthetic images of four ImageNet subsets.

which includes ImageNette, ImageFruit, ImageYellow, and ImageSquawk subdatasets. Within each column, individual images represent specific classes from these subsets. Figure 13 offers a glimpse into the synthetic data generated for six domains within the DomainNet dataset. Similar to the ImageNet visualization, each pair of rows represents one of these domains, which encompasses sketch, real, quickdraw, painting, infograph, and clippart domains. Within each column, you will find synthetic images representing individual classes within the respective domain. Upon close examination, it becomes readily apparent that the synthetic data demonstrates striking color accuracy, precise delineation of object boundaries, and an impressive level of realism that closely approximates that of genuine real-world images. The synthetic images not only maintain vivid and faithful color representations but also capture intricate details, ensuring that the synthetic data closely mirrors the characteristics found in authentic visual data.

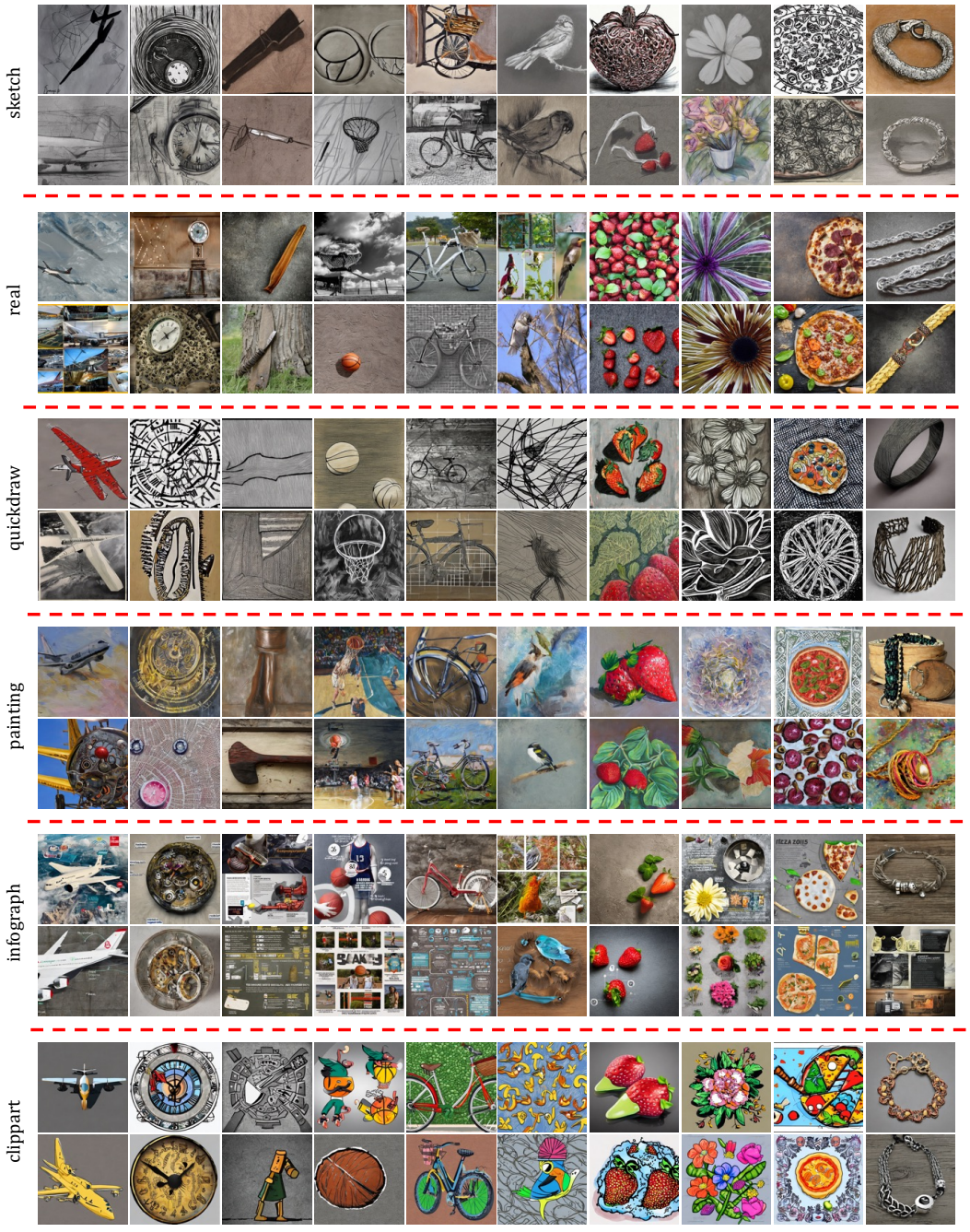


Figure 13: Visualization of synthetic images of six domains from DomainNet dataset.

Table 8: Results for more baselines.

Method	FedAvg	FedOpt	Moon	Ours (one-shot)	Ours (5-round)
ImageNette	72.01	73.21	74.27	85.21	93.80
ImageNet100	40.13	41.25	41.43	48.31	72.67

Table 9: Results on challenging datasets.

Setting		FedAvg			Ours (one-shot)	Ours (5-round)			Centralized
prompt	Dataset	$\beta = 0.01$	$\beta = 0.5$	FedAvg (IID)		$\beta = 0.01$	$\beta = 0.5$	IID	
class	EuroSAT	43.94	74.48	84.87	38.37	37.59	82.94	91.01	94.3
instance					42.7	52.67	92.28	95.26	
class	COVID-19 X-rays	57.0	86.25	95.25	52.0	50.0	70.5	88.75	94.5
instance					54.75	50.0	64.0	44.0	
class	BloodMNIST	64.86	83.14	84.2	30.46	27.68	73.49	84.04	84.76
instance					30.54	37.1	78.78	83.59	
class	DermaMNIST	66.65	69.81	71.22	34.16	45.77	64.75	73.29	71.26
instance					66.79	66.52	69.72	73.59	

6.2.4 Results on more challenging datasets

Even for particularly challenging domains such as remote sensing images or fine-grained classification datasets, our method can easily adapt to these scenarios. We conducted experiments on several fine-grained image classification datasets, namely CUB-200 [53], Stanford Cars [19], and also the satellite image dataset EuroSAT [15]. CUB-200 is a challenging dataset consisting of 200 bird species, while Stanford Cars contains 16,185 images belonging to 196 classes of cars. The size of fine-grained recognition datasets is typically smaller compared to general image classification datasets. In previous work [66, 9], a common practice is to utilize a pretrained model that has been trained on the ImageNet dataset. In this study, we present two approaches: training the model from scratch and loading a pretrained ResNet34 model. As shown in Table 9, our method achieves excellent performance even in these challenging domains. Additionally, in the cross-silo federated learning scenario, when clients have strong computational capabilities, one can simply finetune the foundation models on these domains, achieving better performance than normal federated learning methods.

Class-level Prompts versus Instance-level Prompts. In main experiments, we synthesize a number of samples based on class-level prompts, *e.g.*, *A photo of a [class name]*. Then following [21], we caption individual images with foundation models, *e.g.*, BLIP-2 [25], and synthesize new images based on the individual caption. These prompts provide more precise guidance to the generative model by considering the specific characteristics and context of each sample. Note that [64] also produce instance-level prompts by diffusion inversion, while the inverted samples will leak data privacy and it is very expensive to implement diffusion inversion on large datasets for clients. In Figure 11 (Appendix), we present the performances of two kinds of prompts, where we synthesize 1300 images per class, the same number as the real training set. It is evident that employing a more precise instance-level prompt leads to higher accuracy, *e.g.*, achieving 78.62% on ImageNette, compared to class-level prompt, which achieves 73.20%. This result clearly highlights the importance of considering more detailed instance-level information in prompt design.